Пермский филиал федерального государственного автономного образовательного учреждения высшего образования «Национальный исследовательский университет

«Высшая школа экономики»

Факультет экономики, менеджмента и бизнес-информатики

Образовательная программа бакалавриата «Программная инженерия»

**ОТЧЕТ  
по учебной практике**

Выполнил студент группы ПИ-19-1 Мамедов Нурлан Рагим оглы

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

Проверил:

Руководитель практики   
доцент кафедры информационных

технологий в бизнесе

Бузмаков Алексей Владимирович

\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(оценка) (подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата)

Пермь, 2020 год

# Аннотация

В данном отчете описаны этапы проектирования, разработки и тестирования 12 задач. Программы реализованы на языке высокого уровня C#.

Работа выполнена студентом НИУ ВШЭ Пермь Мамедовым Нурланом Рагим оглы. Кафедра информационных технологий в бизнесе.

Работа содержит 41 страниц формата А4 основного текста, включая 12 глав соответствующей каждой учебной задаче.

Библиографический список включает в себя 4 источника.

В каждой главе содержится анализ задачи (установка функциональных и не функциональных требований к задаче), проектирование алгоритмов и тестирования ПО. Приложения к задачам содержат реализацию задач.

# Введение

Главной задачей данной работы является закрепление и развитие практических навыков в области программирования, алгоритмизации при помощи современных средств разработки, а также в области аналитической и проектной деятельности.

Целью данной работы является закрепление знаний и навыков, полученных по дисциплинам «Дискретная математика», «Компьютерный практикум по основам алгоритмизации и методам программирования», «Линейная алгебра», «Программирование», «Введение в программную инженерию», а также приобретение опыта алгоритмизации задач.

Для достижения поставленной цели необходимо:

1. Наличие практических навыков работы с языком программирования высокого уровня C# и современных сред разработки для реализации предложенных программ.
2. Навык работы с системой контроля версий посредством использования веб-сервиса GitHub.
3. Проведение тестирования разработанных систем и алгоритмов по критериям черного ящика.
4. Развитие практических навыков применения объектно-ориентированного программирования.

Основная часть данной работы содержит результаты выполнения 12 заданий учебной практики и включает в себя: постановку задачи, формат входных и выходных данных, описание решения задачи, разработка функциональных и нефункциональных требований, разработку алгоритмов, процесса реализации и результатов тестирования программы.

# Задача 1

Постановка задачи:

*«Вы работаете в фирме, занимающейся разработкой компьютерных игр. Сейчас вы занимаетесь разработкой новой компьютерной игры "Атака летающих тарелок". По сюжету игры на планету Зумла приземляются летающие тарелки, и их надо уничтожать. Игрок управляет лазерной пушкой. Для того, чтобы произвести выстрел он указывает две точки на поверхности Зумлы (которая в игре считается плоской), через которые должен проходить лазерный луч (который является прямой).*

*Вы должны написать программу, определяющую, какие летающие тарелки были уничтожены выстрелом»*

Входные данные: Первая строка входного файла INPUT.TXT содержит целое число n (1 ≤ n ≤ 30000) - число приземлившихся летающих тарелок. Вторая строка содержит числа xp1, yp1, xp2, yp2 - координаты точек, через которые проходит лазерный луч. Далее идут n строк, каждая из которых содержит описание одной летающей тарелки в формате xi yi ri, где xi, yi - координаты центра, ri - радиус тарелки. Все числа целые и не превосходят по модулю 10000. Радиусы летающих тарелок - целые и положительные. Летающие тарелки могут касаться и пересекаться.

Выходные данные: В первую строку выходного файла OUTPUT.TXT выведите количество уничтоженных летающих тарелок. Во вторую строку выведите номера уничтоженных летающих тарелок в возрастающем порядке. Тарелка считается уничтоженной, если она имеет, хотя бы одну общую точку с лазерным лучом.

Программа должна выполняться не более чем за 1 секунду и требовать не более чем 16 МБ памяти.

Необходимо написать программу, которая высчитывала бы перекрытие «лазера» (прямой) и «летающих тарелок» окружностей на плоскости. Заданы количество окружностей, две точки, через которые проходит прямая, точки центров окружностей и соответствующие радиусы.

## Анализ

Для решения задачи нужно:

1. Составить уравнение прямой (лазера) по двум точкам в виде:

(1.1)

Где A, B, C выводятся из формулы уравнения прямой, проходящей через две заданные точки на плоскости:

(1.2)

Тогда:

(1.3)

1. Расстояние от точки до прямой равно:

(1.4)

1. Теперь требуется пройти циклом по всем окружностям и записывать в ответ те, у которых радиус меньше или равен расстоянию от точки до прямой.

## Проектирование

Ниже представлена блок-схема, составленная на основе вышеописанного анализа и приведенного алгоритма.

Завершение

Нет

Да

Записывает тарелку в результат

Если |A \* x + B \* y + C| / Math.Sqrt(A \* A + B \* B) меньше радиуса тарелки

Проходит по всем координатам тарелок

A = y2 – y1

B = x1 – x2

C = x2 \* y1 – x1 \* y2

Считывание количества тарелок, координат лазера, координат центра тарелок и их радиуса

Начало

## Тестирование



Рисунок 1.1 Тестирование задачи 1

Требования к скорости работы и используемой памяти были выполнены.

# Задача 2

Петя разгадывает головоломку, которая устроена следующим образом. Дана квадратная таблица размера N×N, в каждой клетке которой записана какая-нибудь английская буква. Кроме того, дан список ключевых слов. Пете нужно, взяв очередное ключевое слово, найти его в таблице. То есть найти в таблице все буквы этого слова, причем они должны быть расположены так, чтобы клетка, в которой расположена каждая последующая буква слова, была соседней с клеткой, в которой записана предыдущая буква (клетки называются соседними, если они имеют общую сторону — то есть соседствуют по вертикали или по горизонтали). Например, на рисунке показано, как может быть расположено в таблице слово olympiad.

Когда Петя находит слово, он вычеркивает его из таблицы. Использовать уже вычеркнутые буквы в других ключевых словах нельзя. После того, как найдены и вычеркнуты все ключевые слова, в таблице остаются еще несколько букв, из которых Петя должен составить слово, зашифрованное в головоломке.

Помогите Пете в решении этой головоломки, написав программу, которая по данной таблице и списку ключевых слов выпишет, из каких букв Петя должен сложить слово, то есть какие буквы останутся в таблице после вычеркивания ключевых слов.

Входные данные: Во входном файле INPUT.TXT записаны два числа N (1 ≤ N ≤ 10) и M (0 ≤ M ≤ 100). Следующие N строк по N заглавных английских букв описывают ребус. Следующие M строк содержат слова. Слова состоят только из заглавных английских букв, каждое слово имеет длину от 1 до 100 символов. Гарантируется, что в таблице можно найти и вычеркнуть по описанным выше правилам все ключевые слова.

Выходные данные: В выходной файл OUTPUT.TXT выведите в алфавитном порядке оставшиеся в таблице буквы.

Программа должна выполняться не более чем за 1 секунду и требовать не более чем 16 МБ памяти.

## Анализ

Изначально может показаться, что для решения задачи, придётся реализовывать весь алгоритм вычёркивания слов из таблицы. Но при тщательном чтении обнаруживается, что порядок букв в ответе совершенно неважен. Отталкиваясь от этого, для решения задачи можно составить следующий алгоритм:

1. Считать входные данные.
2. Записать все слова таблицы в одну строку.
3. Записать все ключи в одну строку.
4. Пройти по каждому символу ключей.
5. Удалять вхождения символа ключей в строку слов.
6. Отсортировать оставшиеся буквы в строке слов в алфавитном порядке.

## Проектирование

По вышеописанному алгоритму составлена данная блок-схема.

Начало

Считывание входных данных

Запись всех слов таблицы в одну строку

Запись всех ключей в одну строку

Есть ли символ в строке слов таблицы?

Нет

А

Вывод ответа

Завершение

Проходит по каждому символу строки ключей

Да

А

Удаление символа из строки слов таблицы

## Тестирование



Рисунок 1..2 Тестирование задачи 2

Требования к скорости работы и используемой памяти были выполнены.

# Задача 3

Даны действительные числа x, y. Определить, принадлежит ли точка с координатами x, y заштрихованной части плоскости.

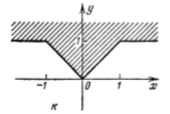


Рисунок 1.3

## Анализ

Задача являет собой проверку на принадлежность точки графикам:

(1.5)

Для решения задачи необходимо:

1. Считать координаты точки.
2. Проверить на соответствие условию.
3. Вывести результат.

Местом потенциальной ошибки является ввод координатов. Поэтому требуется дополнительно написать метод для проверки ввода.

## Проектирование

Согласно проведенному анализу, были составлены блок-схемы для методов.

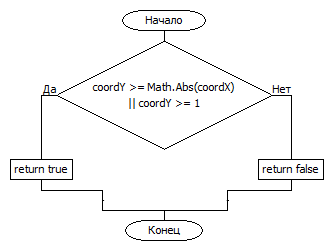


Рисунок 1.4 Блок-схема для задачи 3

Дополнительно была создана перегрузка метода для проверки принадлежности точки с одной координатой (координата по оси абсцисс принимается за нуль).

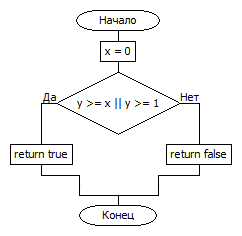


Рисунок 1.5 Блок-схема проверки на принадлежность с одним параметром

Также был спроектирован метод для проверки ввода пользователя.

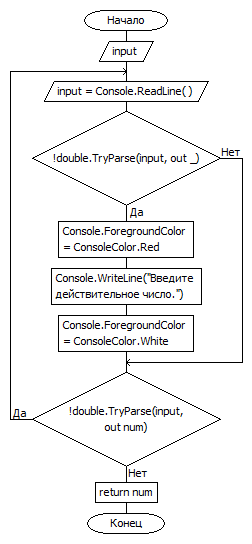


Рисунок 1.6 Блок-схема проверки ввода пользователя

## Тестирование

Для тестирования был составлена таблица тестов - чёрный ящик. Таблица содержит критерии проверки, входные данные и ожидаемые выходное данные.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Критерии черного ящика** | **Тесты** | | | |
| Критерий | Т1 | Т2 | Т3 | Т4 |
| **1. Значение X** |  | | | |
| 1.1 Равен нулю | + |  |  |  |
| 1.2 Единица |  | + |  |  |
| 1.3 Больше единицы |  |  | + |  |
| **2. Значение Y** |  | | | |
| 2.1 Равен нулю | + |  | + |  |
| 2.2 Единица |  | + |  |  |
| 2.3 Больше единицы |  |  |  | + |
| **3. Принадлежность графику** |  | | | |
| 3.1 Принадлежит выделенной области | + | + |  | + |
| 3.2 Не принадлежит выделенной области |  |  | + |  |
| 3.3 Точка лежит на границе графика | + | + |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Номер теста** | **Входные данные** | **Ожидаемые выходные данные** | **Примечание** |
| 1 | x = 0 y = 0 | Точка (0, 0) принадлежит графику | 1.1, 2.1, 3.1, 3.3 |
| 2 | x = 1 y = 1 | Точка (1, 1) принадлежит графику | 1.2, 2.2, 3.1, 3.3 |
| 3 | x = 2 y = 0 | Точка (2, 0) не принадлежит графику | 1.3, 2.1, 3.2 |
| 4 | x = -1 y = 2 | Точка (-2, 2) принадлежит графику | 2.3, 3.1 |

# Задача 4

Дано действительное положительное число ε. Методом хорд вычислить с точностью ε\*) корень уравнения f(x) = 0.

## Анализ